AiU Certified Tester in AI (CTAI)
Programa de estudio

Versión 1.01R 2019

Artificial Intelligence United

Aviso de derechos de autor

Gracias a los autores principales:

- Vipul Kocher, Saurabh Bansal, Srinivas Padmanabhuni and Sonika Bengani

Gracias a los coautores

- Rik Marselis and José M. Díaz Delgado

Gracias al comité de revisión

Albert Tort, Alfonso Fernández, Amit Dang, Ana Laura Ochoa Moreno, Andreas Hetz, Ángel Rayo Acevedo, Aurelio Gandarillas, Baris Sarialioglu, Björn Lemke, Christine Green, Daniel Garcia Castillo, Daniel Tolosa, Dario Iván Rosas Miranda, Durga Mohapatra, Emilie Potin-Suau, Erik van Veenendaal, Girish Nuli, Girts Baltaisbrencis, Guino Henostroza, Gustavo Márquez Sosa, Gustavo Terrera, Héctor Ruvalcaba, Javier Alejandro Chávez Crivelli, Jeff Nyman, Joel

Oliveira, José Antonio Rodríguez Gómez, Juan Pablo Rios Alvarez, Julie Gardiner, Kimmo Hakala, Kristine Corbus, Kyle Alexander Siemens, Lorena Parra Rubio, Maarten-Jan van Gool, Manuel Fischer, Márton Görög, Maximiliano Mannise, Melissa Pontes, Miaomiao Tang, Michaël Pilaeten, Michel Dussouchaud, Nadia Soledad Cavalleri, Nora Alriyes, Paweł Noga, Petr Neugebauer, Ralf Pichler, Ram Shanmugam, Raúl Hussein Galindo, Richard Seidl, Sammy Kolluru, Samuel Ouko, Santiago de Jesús González Medellín, Sebastia Małyska, Sergio Emanuel Cusmai, Serge Wolf, Shantel Yanique Stewart, Silvia Nane, Sunil Godse, Siva Prasad. B, Søren Wassard, Tariq King, Tetsu Nagata, Vikas Dhaka, Tom Van Ongeval, Werner Lieblang, Wim Decoutere, Yogesh Ahuja, Young jae Choi

Historial de revisiones

Versión	Fecha	Observaciones
AiU A 2019	2 de marzo 2019	Primera versión beta
AiU B 2019	8 de mayo 2019	Segunda versión beta
AiU 1.0R 2019	16 de julio 2019	Primera publicación
Versión 1.01R 2019	3 de septiembre, 2019	Secunda publicación

Índice

Objetivos de Negocio

BO-1	Comprender las tendencias actuales, las aplicaciones industriales de la Inteligencia Artificial (AI) utilizando el Aprendizaje Automático (Machine Learning - ML).
BO-2	Comparar los diferentes algoritmos ML implementados para ayudar a elegir el más adecuado.
BO-3	Evaluar modelos para el aprendizaje supervisado y no supervisado.
BO-4	Diseñar y ejecutar casos de prueba para sistemas de AI.
BO-5	Utilizar varios métodos para aportar transparencia en el funcionamiento de los modelos
BO-6	Definir una estrategia de prueba para probar sistemas de AI.
BO-7	Comprender dónde se puede utilizar la AI en las pruebas manuales y en la automatización de pruebas.
BO-8	Utilizar herramientas de ejecución de pruebas basadas en la AI para automatizar las pruebas.

Objetivos de aprendizaje/niveles cognitivos de conocimiento

Los objetivos de aprendizaje (Learning objectives, LO) son enunciados breves que describen lo que se espera que sepa después de estudiar cada capítulo. Las LOs se definen en base a la taxonomía modificada de Bloom, de la siguiente manera:

- K1: Recordar. Algunos de los verbos de acción son Recordar, Rememorar, Elegir, Definir, Buscar, Emparejar, Relacionar, Seleccionar
- K2: Entender. Algunos de los verbos de acción son Resumir, Generalizar, Clasificar, Comparar, Contrastar, Demostrar, Interpretar, Reformular
- K3: Aplicar. Algunos de los verbos de acción son Implementar, Ejecutar, Usar, Aplicar

Para más detalles de la taxonomía de Bloom, consulte [BT1] y[BT2] en Referencias.

Objetivos prácticos

Los Objetivos Prácticos (Hands-on Objectives, HOs) son enunciados breves que describen lo que se espera que realice o ejecute para entender el aspecto práctico del Aprendizaje.

Los HOs se definen de la siguiente manera:

- HO-0: Demostración en vivo de un ejercicio o video grabado

- HO-1: Ejercicio guiado. Los alumnos siguen la secuencia de pasos que realiza el formador
- HO-2: Ejercicio con sugerencias — Ejercicio a ser resuelto por los participantes utilizando las sugerencias proporcionadas por el capacitador
- HO-3: Ejercicios no guiados sin sugerencias

Prerrequisitos

Obligatorio

- Ninguno

Recomendado

- ISTQB® nivel Fundamentos (CTFL) o equivalente
- Conocimientos básicos de cualquier de los lenguajes de programación siguientes - Java/Python/R
- Conocimientos básicos de estadística
- Alguna experiencia en desarrollo de software o pruebas

Capítulo 1 - Introducción a la Inteligencia Artificial (AI)

Palabras clave

inteligencia artificial (Artificial Intelligence, AI), aprendizaje automático (Machine Learning, ML), aprendizaje supervisado, aprendizaje no supervisado, clasificación supervisada, regresión supervisada, agrupamiento no supervisado, asociación no supervisada, aprendizaje por refuerzo (Reinforcement Learning, RL), aprendizaje profundo (Deep Learning, DL), redes neuronales (Neural Networks, NN), CRISP-DM, ciclo de vida de ML.

LO-1.1.1	K2	Explicar la inteligencia artificial (AI).
LO-1.1.2	K1	Recordar varios tipos de AI — Específica, General y Súper AI.
LO-1.2.1	K2	Explicar el aprendizaje automático (ML) y el hecho de que el ML es una manera de lograr la AI.
LO-1.2.2.	K2	Explicar el ML Supervisado y la diferencia entre Clasificación Supervisada y Regresión Supervisada.
LO-1.2.3.	K2	Explicar el aprendizaje automático (ML) sin supervisión y comparar el agrupamiento (clustering) y la asociación no supervisados.
LO-1.2.4.	K1	Recordar la definición y las aplicaciones del Aprendizaje por Refuerzo (Reinforcement Learning, RL).
LO-1.3.1.	K2	Explicar el aprendizaje profundo (Deep Learning, DL) y los tipos de redes neuronales (Neural Networks, NN).
LO-1.4.1	K2	Explicar varias etapas de CRISP-DM — un proceso para el ciclo de vida de ML.

LO-1.4.2	K3	Aplicar los pasos necesarios para identificar el tipo de problema de ML apropiado.

1.1 Inteligencia Artificial (AI)

La inteligencia artificial (Artificial Intelligence, AI) es la inteligencia proporcionada por una máquina con el objetivo de resolver problemas normalmente resueltos con inteligencia humana.

Actualmente, existen ya numerosas aplicaciones basadas en AI. Algunos ejemplos son IBM Watson, MS Cortana, Siri de Apple, los vehículos autónomos, etc.

1.1.1 Definición de Inteligencia Artificial (AI)

LO-1.1.1	K2	Explicar la inteligencia artificial (AI).

La AI es el arte de crear máquinas que realizan funciones que requieren inteligencia cuando son realizadas por personas. [KUR] La AI está teniendo un papel protagonista en sectores como la salud, la manufactura, el comercio electrónico, las redes sociales, la logística, y otros sectores industriales. Algunas de las razones del uso creciente de la AI son el aumento de la potencia de procesamiento, la disponibilidad de datos y las nuevas tecnologías. Los casos de uso de la AI van desde el monitoreo de las viviendas, la determinación de las acciones en las que se debe invertir, la ayuda para decidir qué receta se debe hacer, ¡hasta la ayuda para elegir a la pareja para toda la vida!

AI es un término genérico que abarca la ciencia de hacer que las máquinas sean inteligentes, ya sea un robot, un refrigerador, un televisor, un auto, un firmware o un componente de software. El aprendizaje automático (Machine Learning, ML), es el subconjunto de AI. El ML y la AI a menudo se usan indistintamente, pero no son lo mismo.

ML se explica con más detalle en 1.2 Aprendizaje automático (Machine Learning, ML)

1.1.2 Tipos de AI

LO-1.1.2	K1	Recordar varios tipos de AI — Específica, General y Súper AI.

La AI puede clasificarse en general como Específica, General o Súper AI.

- **AI Específica (débil)**: Máquinas que están programadas para llevar a cabo una tarea específica con un contexto limitado. Por ejemplo, máquinas recreativas, asistentes de voz y toda la AI actual.
- **AI General**: Las máquinas con capacidades cognitivas generales se denominan popularmente casos de AI Fuerte. Estas AIs pueden razonar y entender su entorno como lo hacen los humanos, y actuar en consecuencia. Por ejemplo, el razonamiento de sentido común. Actualmente, la AI General no se ha llevado a cabo, y nadie sabe si se hará en realidad o cuándo.
- **Súper AI**: Máquinas que son capaces de replicar pensamientos, ideas y emociones humanas. Es ese súper estado de inteligencia en el que las máquinas se volverán más inteligentes y sabias que los humanos. Considerando el estado actual de los desarrollos de la AI, Super AI no se convertirá en una realidad en un futuro cercano.

1.2 Aprendizaje automático (Machine Learning, ML)

1.2.1 Definición de ML

LO-1.2.1	K2	Explicar el aprendizaje automático (ML) y el hecho de que el ML es una manera de lograr la AI.

Arthur Samuel definió el aprendizaje automático, conocido con el término Machine Learning (ML) como “un campo de estudio que da a las computadoras la capacidad de aprender sin estar explícitamente programadas.” Los sistemas de ML aprenden y mejoran con la experiencia y, con el tiempo, refinan un modelo que puede ser usado para predecir el resultado de las preguntas, basado en el aprendizaje previo [JB1].

Más allá del ML, la AI utiliza conceptos de representación del conocimiento y razonamiento para manejar diversos escenarios. Las nociones de búsqueda, programación y optimización caen dentro del ámbito de la AI, pero no necesariamente del ML.

Algunas de las tecnologías utilizadas para lograr la AI son:

- Aprendizaje automático (Machine Learning, ML)
- Procesamiento del lenguaje natural (Natural Language Processing, NLP)
- Robótica
- Procesamiento de voz
- Visión por computador (Computer Vision)

Hay algunas maneras en las que los algoritmos ML pueden ser categorizados:

- Aprendizaje supervisado
- Aprendizaje no supervisado
- Aprendizaje por Refuerzo (Reinforcement Learning, RL)

1.2.2 Aprendizaje supervisado - Clasificación y regresión

LO-1.2.2	K2	Explicar el ML Supervisado y la diferencia entre Clasificación Supervisada y Regresión Supervisada.

Aprendizaje supervisado: En este tipo de aprendizaje, el modelo aprende de datos etiquetados durante la fase de formación. Los datos etiquetados actúan como un entrenador/supervisor para la función de mapeo que infiere la relación entre los datos de entrada y la etiqueta de salida durante la capacitación. Durante la fase de prueba, la función de mapeo se aplica a un nuevo conjunto de datos no usados para la capacitación para predecir la salida que también está etiquetada. El modelo se despliega una vez que el nivel de precisión de salida es satisfactorio.

Los problemas resueltos por el Aprendizaje Supervisado se dividen a su vez en dos categorías:

Clasificación: Cuando el problema requiere clasificar un dato en una de las pocas clases predefinidas, se utiliza el aprendizaje supervisado. Este tipo de modelo se utiliza cuando los datos de salida son discretos o cuando la salida cae

entre el número de clases alimentadas durante el entrenamiento. El reconocimiento facial o la detección de objetos en una imagen son ejemplos de problemas que pueden utilizar la clasificación. Otras aplicaciones de la clasificación son la detección de spam (spam o no spam), el diagnóstico de una enfermedad sobre la base por ejemplo de una radiografía, la identificación correcta de las señales de tráfico por parte de un sistema de asistencia al conductor, etc. Algunos de los algoritmos comúnmente utilizados para la clasificación son la regresión logística, el vecino más cercano, la máquina vectorial de soporte (Support vector machine, SVM) y las redes neuronales (Neural Networks, NN).

Regresión: Cuando los datos de salida son de naturaleza continua o numérica, por ejemplo, prediciendo la edad/el peso de una persona, prediciendo el precio futuro de la acción, etc., se utiliza el aprendizaje de regresión. El algoritmo más utilizado para este tipo de problemas es la regresión lineal, un algoritmo simple que explica la relación entre las entradas y las salidas como una ecuación lineal. Otros algoritmos son la regresión logística, las máquinas vectoriales de soporte, la regresión Lasso, etc.

1.2.3 Aprendizaje no supervisado - Agrupación y asociación

LO-1.2.3	K2	Explicar el aprendizaje automático (ML) sin supervisión y comparar el agrupamiento (clustering) y la asociación no supervisados.

Aprendizaje no supervisado: Los datos limpios y etiquetados no están disponibles todo el tiempo, por lo que ciertos problemas deben ser resueltos sin un conjunto de entrenamiento explícitamente etiquetado. Este tipo de ML en el que no se proporcionan datos etiquetados explícitamente se denomina Aprendizaje no supervisado. El objetivo en tales problemas es aprender el patrón y la estructura de los datos de entrada sin ninguna etiqueta asociada.

El aprendizaje no supervisado se clasifica además en los dos métodos siguientes, según el tipo de salidas:

Agrupamiento: Este modelo de aprendizaje no supervisado agrupa los datos de entrada basándose en características o atributos comunes. Los datos de entrada con atributos similares (no etiquetados) se agrupan de manera automática. Por lo tanto, el resultado obtenido son grupos de datos de entrada con

características similares. Un ejemplo ilustrativo es la segmentación de clientes que se puede realizar en el análisis de mercados.

Asociación: La minería de reglas de asociación encuentra relaciones o dependencias interesantes entre los atributos de datos. El descubrimiento de asociaciones interesantes proporciona una fuente de información que a menudo se utiliza para la toma de decisiones. Por ejemplo, el análisis de datos del carrito de compras, el sistema de recomendación de productos basado en los aprendizajes derivados del comportamiento de compra del cliente son buenos ejemplos del modelado basado en reglas de asociación.

1.2.4 Aprendizaje por Refuerzo (RL)

LO-1.2.4	K1	Recordar la definición y las aplicaciones del Aprendizaje por Refuerzo (Reinforcement Learning, RL).

Es un tipo de ML en el que un agente (algoritmo) aprende interactuando con el entorno de forma iterativa y, por lo tanto, aprende de la experiencia. El agente es recompensado cuando toma una decisión correcta y penalizado cuando toma una decisión equivocada. Este aprendizaje, basado en recompensas y sanciones, se define así como “aprendizaje de refuerzo” (Reinforcement learning, RL). Establecer el ambiente adecuado, elegir la estrategia correcta para que el agente cumpla con la meta deseada y diseñar una función de recompensa, son algunos de los desafíos clave en la implementación de RL. Robótica, Vehículos Autónomos y Chatbots son ejemplos de aplicaciones que pueden usar RL.

1.3 Aprendizaje profundo (Deep Learning,DL)

1.3.1 El aprendizaje profundo (DL) y los tipos de redes neuronales (Neural Networks, NN)

LO-1.3.1	K2	Explicar el aprendizaje profundo (Deep Learning, DL) y los tipos de redes neuronales (Neural Networks, NN).

El aprendizaje profundo (Deep Learning, DL) se refiere a los sistemas que obtienen experiencia de conjuntos de datos masivos. DL utiliza redes neuronales artificiales (Artificial Neural Networks, ANN) para analizar grandes

conjuntos de datos, por ejemplo, vehículos autónomos, procesamiento de texto de gran tamaño y aplicaciones de visión por computador, entre otros. [AG1]

El DL es un subconjunto del ML y el ML es un subconjunto de la AI. El DL utiliza los mismos tipos de aprendizaje (Aprendizaje Supervisado, No Supervisado y de Refuerzo, RL) que el ML.

Redes Neuronales Artificiales: Las redes neuronales artificiales (Artificial Neural Networks, ANN) se inspiran en la arquitectura del cerebro humano. Las "neuronas", como unidad básica de la ANN, actúan sobre el estímulo de entrada y producen la señal de salida. La entrada pasa por las capas de funciones de activación para generar la salida. Estas capas forman una red en forma de malla.

Cada ANN tiene al menos dos capas — capas de entrada y de salida. Todas las capas entre estas dos capas se llaman capas ocultas. Algunos de los varios tipos de redes neuronales (NN) son:

Red Neural Profunda (DNN): La Red Neural Profunda (Deep Neural Network, DNN) es una red neuronal artificial (ANN) con dos o más capas ocultas.

Red Neuronal Convolucional (Convolutional Neural Network, CNN): La Red Neuronal Convolucional (CNN) es una ANN que surgió del estudio de la corteza visual del cerebro, y se ha utilizado en el reconocimiento de imágenes desde la década de 1980. A diferencia de otras redes neuronales (NN), las CNNs trabajan directamente en las imágenes de entrada sin serializar/vectorizar una imagen de entrada y extrayendo las características mediante filtros. Las CNNs proporcionan servicios de búsqueda de imágenes, vehículos autónomos, sistemas de clasificación automática de vídeo y más.

Red Neuronal Recurrente (Recurrent Neural Network, RNN): Estas ANNs pueden predecir el futuro de los problemas de las series temporales. Siguen un enfoque secuencial en series de datos de entrada de longitud arbitraria en lugar de entradas de longitud fija como en otras redes neuronales (NN). Cada entrada y salida son independientes de todas las demás capas. La retroalimentación de la capa de salida se alimenta a la misma red de forma recurrente, hasta que se alcanza el nivel de confianza adecuado. Las RNNs pueden analizar datos de series de tiempo, tales como los precios de las acciones, y decirle cuándo comprar o vender. En vehículos autónomos, pueden anticipar trayectorias y ayudar a evitar accidentes.

1.4 Etapas del proceso de ML

Un proyecto ML típico sigue todas las etapas del proceso estándar de la industria híbrido para la minería de datos (CRISP-DM) — un estándar de la industria, y un marco flexible.

1.4.1 Etapas del Proceso ML — Proceso CRISP-DM

LO-1.4.1	K2	Explicar varias etapas de CRISP-DM — un proceso para el ciclo de vida de ML.

CRISP-DM tiene tradicionalmente seis etapas en el ciclo de vida de la minería de datos. Se ha personalizado para cumplir con los requisitos de los proyectos de ML, añadiendo una séptima etapa.

Las siete etapas del marco CRISP-DM para el ML son: [DSC1][SMU]

1. **Adquisición de datos**: Recopilar datos de todas las fuentes internas y externas (por ejemplo, bases de datos, archivos CSV, medios sociales, etc.)

2. **Preparación de datos**: Limpiar los datos sin procesar y remodelarlos. Se crean nuevos atributos con la ingeniería de características, un proceso para crear nuevas variables a partir de los datos existentes. La reducción de la dimensionalidad, la imputación de datos, el tratamiento del valor nulo de los valores que faltan, etc., son algunos de los métodos que intervienen en la preparación de los datos.

3. **Modelado**: Seleccionar el modelo o algoritmo, dividir los datos disponibles en conjunto de entrenamiento y conjunto de pruebas. Los modelos se obtienen ejecutando algoritmos ML en el conjunto de datos de entrenamiento. Utilizar el conjunto de datos de pruebas para evaluar y mejorar el rendimiento del modelo hasta que se logre un rendimiento satisfactorio.

4. **Evaluación**: valuar el modelo en varias métricas (discutidas en 3.2 Métricas) y establecer la línea de base antes de su despliegue final.

5. **Despliegue**: Desplegar y monitorear el modelo de línea base para las métricas en el entorno de producción.

6. **Operaciones**: Llevar a cabo el mantenimiento y las operaciones regulares. Regenerar y refinar el modelo cuando las métricas caen por debajo de un umbral determinado.

7. **Optimización**: La solución desplegada puede ser reemplazada debido a la derivada del concepto (ver 6.1.3 Riesgo de deriva del concepto (Concept Drift, CD), a medida que se disponga de mejores algoritmos, o debido a algunos fallos importantes en el rendimiento.

Los pasos 1-4 se pueden clasificar como parte de la fase fuera de línea (offline), cuyo resultado es el modelo entrenado. Los pasos 5-6 forman parte de la fase en línea (online), donde el modelo formado en la fase offline se integra con el resto del sistema y se despliega en un entorno de producción. El paso de optimización implica ejecutar nuevamente los pasos 1-6.

1.4.2 Pasos para la identificación del tipo de problema de ML

LO-1.4.2	K3	Aplicar los pasos necesarios para identificar el tipo de problema de ML apropiado.

Es importante entender los problemas que estamos tratando de resolver y el tipo de aprendizaje que se requiere para resolverlos. A continuación, se analiza una forma de identificar el tipo de problema de ML:

1. Si el problema involucra la noción de múltiples estados, e involucra movimientos en cada estado, entonces explore RL.
2. Si hay una variable de salida, se trata de un aprendizaje supervisado.
   2.1. En el caso de que la salida sea discreta y categórica, es un problema de clasificación.
   2.2. En el caso de que la salida sea de naturaleza numérica y continua, entonces es un problema de regresión.
3. Si la salida no se proporciona en el conjunto de datos dado, entonces explore el aprendizaje sin supervisión.
   3.1. Si el problema consiste en agrupar datos similares, entonces se trata de un problema de agrupamiento (cluster).
   3.2. Si el problema consiste en encontrar elementos de datos con coocurrencia, entonces aplique la minería de reglas de asociación.
   3.3. Si los datos brutos no están estructurados, las funciones de extracción automática se pueden explorar automáticamente con algoritmos de Deep Learning (DL).

El requisito previo a los pasos anteriores es que debe haber suficientes datos disponibles para analizar el tipo de problema de ML adecuado.

Capítulo 2 - Descripción general de las pruebas de los sistemas de AI

Palabras clave: fase fuera de línea (offline), fase en línea (online), ISO25010

LO-2.1.1	K2	Explicar las pruebas realizadas en sistemas AI-ML en la fase de formación (offline) y en la fase de integración post-formación (online).
LO-2.2.1	K2	Compare las pruebas de los sistemas de AI con los sistemas que no son de AI.
LO-2.3.1	K1	Recordar las características de calidad ISO 25010, especialmente la idoneidad funcional, la eficiencia de rendimiento, la fiabilidad, la mantenibilidad y otros parámetros como la complejidad, la escalabilidad y el aprendizaje continuo como parámetros a utilizar para la evaluación del modelo de formación.
LO-2.3.2	K1	Recordar las características de calidad específicas de las pruebas de inteligencia artificial, además de las mencionadas en la norma ISO25010 — comportamiento inteligente, moralidad y personalidad.

2.1 Fases de prueba de la AI

2.1.1 Pruebas en línea (online) y fuera de línea (offline) de sistemas de AI

LO-2.1.1	K2	Explicar las pruebas realizadas en sistemas AI-ML en la fase de formación (offline) y en la fase de integración post-formación (online).

El ciclo de vida del ML, tal como se describe en 1.4 Etapas del Proceso de ML, puede dividirse en dos fases: fuera de línea (offline) y en línea (online). Los distintos tipos de pruebas que se realizan durante estas fases son:

Pruebas de fase offline: En esta fase se prueba el modelo entrenado. Se utilizan varias métricas para los parámetros de evaluación de un modelo entrenado para verificar hasta qué punto ha alcanzado los objetivos. Existen diferentes parámetros para el aprendizaje supervisado y no supervisado. Típicamente, el modelo se prueba en cuanto a su comportamiento funcional, pero las características no funcionales no se prueban. Dado que el modelo se implementa en un entorno diferente al de la formación, realizar pruebas de rendimiento del modelo en esta fase no tiene mucho sentido. El tiempo de entrenamiento modelo es un parámetro que se evalúa como un parámetro no funcional. Para los modelos que necesitarán ser reentrenados frecuentemente, este puede ser un parámetro importante. Las pruebas offline se tratan en el Capítulo 3 - Pruebas offline de sistemas de AI.

Otro aspecto importante de las pruebas offline es si es posible explicar el comportamiento del modelo. Hay varios métodos y algoritmos que se pueden utilizar para ello. Este aspecto de las pruebas se trata en el Capítulo 5 - Inteligencia Artificial explicable.

Pruebas de fase online: En esta fase, se prueba la integración del modelo entrenado con el resto del sistema, incluyendo todos los demás componentes de AI y no AI. Se pueden realizar tanto pruebas funcionales como no funcionales, tales como pruebas de rendimiento.

Dado que las entradas al sistema pueden ser no textuales, las entradas no estructuradas, así como el soporte de automatización para algunas de las pruebas relacionadas con la parte ML, pueden ser limitadas, en función de la herramienta que se utilice.

Las pruebas online se tratan en el Capítulo 4 - Pruebas online de los sistemas de AI.

2.2 Pruebas de AI vs. Pruebas no de AI

2.2.1 Pruebas de sistemas de AI vs. Sistemas Tradicionales (no AI)

LO-2.2.1	K2	Comparar las pruebas de los sistemas de AI con los sistemas que no son de AI.

- Los oráculos de prueba para los sistemas de AI no son fáciles de conseguir.

- A diferencia de las pruebas tradicionales, los resultados de las pruebas de los sistemas de AI no son deterministas. La salida de un sistema de AI tiene probabilidades, por lo tanto, el resultado de un caso de prueba es también probabilidades en lugar de un aprobado/reprobado definitivo.
- La lógica de los sistemas de AI se genera a partir de los datos utilizados para entrenar el modelo. Esa lógica no está disponible para su evaluación, especialmente las redes neuronales (NN). Esto dificulta la comprensión de por qué se ha producido una determinada salida. Una respuesta correcta o deseada no garantiza el funcionamiento correcto.
- La prueba en la fase offline es una fase adicional que requiere habilidades y técnicas especializadas, como las relacionadas con la limpieza y preprocesamiento de datos, y la prueba del modelo entrenado.
- Las pruebas en la fase online requieren una comprensión profunda de cómo funcionan los sistemas de AI. La integración de sistemas AI con otros sistemas AI y no AI aumenta la necesidad de diferentes técnicas de diseño de pruebas.
- Al igual que con los sistemas que no son de AI, es necesario realizar pruebas funcionales y no funcionales de los sistemas de AI.
- Las pruebas de fase online se pueden realizar como pruebas normales de integración de sistemas y sistemas de caja negra sin preocuparse de si hay uno o más componentes de AI en la mezcla.

2.3 Características de calidad de la AI

2.3.1 Características de calidad para la evaluación de sistemas de AI

LO-2.3.1	K1	Recordar las características de calidad ISO 25010, especialmente la idoneidad funcional, la eficiencia de rendimiento, la fiabilidad, la mantenibilidad y otros parámetros como la complejidad, la escalabilidad y el aprendizaje continuo como parámetros a utilizar para la evaluación del modelo entrenado.

Las características de calidad de un sistema de AI pueden evaluarse mediante una combinación de características de calidad de la norma ISO 25010 y otras características de calidad. Algunas de las características importantes desde la perspectiva de las pruebas de AI son:

- **Idoneidad funcional** - Corrección funcional, integridad e idoneidad.

- **Fiabilidad**
  - Disponibilidad - Disponibilidad del sistema durante las operaciones normales.
  - Tolerancia a fallos - Capacidad del sistema para manejar datos corruptos, incompletos o irrelevantes sin averías.

- **Eficiencia en el rendimiento**
  - Comportamiento temporal - la rapidez con la que el sistema responde a las demandas que se le hacen.
  - Utilización de recursos - Cuáles y cuántos recursos utiliza el sistema para realizar una función.

- **Mantenibilidad**
  - Facilidad de ser analizado - El grado de efectividad y eficiencia con el que es posible evaluar el impacto en un producto o sistema de un cambio previsto en una o más de sus partes, o diagnosticar un producto por deficiencias o causas de fallas, o identificar las partes que han de ser modificadas. En el caso de la AI, la facilidad de ser analizado también se refiere a la capacidad de comprender por qué el sistema tomó la decisión que tomó. Idealmente, la explicabilidad (o examinabilidad) debería ser una característica de calidad separada para ISO 25010 para sistemas de AI.

  - Facilidad para ser probado - El grado en que el componente de AI soporta las pruebas en un contexto determinado. Cuanto más alto sea el grado/la facilidad para ser probado, más fácil será encontrar errores mediante pruebas.

Otros parámetros importantes son:

- **Complejidad** — Complejidad temporal y espacial

- **Escalabilidad** — La capacidad del sistema para manejar una mayor carga mediante la adición de recursos adicionales.

- **Aprendizaje continuo** — La capacidad del sistema para aprender continuamente de los nuevos datos, especialmente de los datos del entorno en tiempo real.

2.3.2 Características extendidas de calidad específicas de la AI

LO-2.3.2	K1	Recordar las características de calidad específicas de las pruebas de inteligencia artificial, además de las mencionadas en la norma ISO25010: comportamiento inteligente, moralidad y personalidad.

El uso de la AI ha requerido una extensión de las características de calidad estándar.

Una máquina inteligente también debe tener las siguientes características de calidad, además de las mencionadas en ISO25010. [TDA]

- Comportamiento inteligente — El comportamiento inteligente es la capacidad de comprender o entender. Es básicamente una combinación de razonamiento, memoria, imaginación y juicio; cada una de estas facultades depende de las demás. La inteligencia es una combinación de habilidades cognitivas y conocimientos que se hacen evidentes por los comportamientos que son adaptables. Las subcaracterísticas son: Capacidad de aprendizaje, Improvisación, Transparencia de las elecciones, Colaboración e Interacción natural.
- La moralidad — La moralidad, en relación con la AI, se refiere a los principios relativos a la distinción entre el bien y el mal o entre el buen y el mal comportamiento. Las subcaracterísticas son: Ética, Privacidad y Amigabilidad Humana.
- Personalidad — La personalidad es la combinación de características o cualidades que forman el carácter distintivo de una persona. Las subcaracterísticas son: Estado de ánimo, empatía, humor y carisma.

Capítulo 3 - Pruebas offline de los sistemas de AI

Palabras clave: datos estructurados, datos no estructurados, dimensionalidad, métricas, validación cruzada, subajuste (underfitting), sobreajuste (overfitting), analítica (analytics).

LO-3.1.1	K1	Recordar los pasos necesarios para la preparación y el preprocesamiento de los datos, así como la necesidad de una limpieza exhaustiva de los mismos.
LO-3.1.2	K3	Aplicar la lectura y manipulación de datos, incluyendo los pasos de filtrado para datos estructurados.
HO-3.1.2	H2	Utilice el código (lenguaje de su elección) para leer los datos de varias fuentes de datos, como un archivo Excel, un archivo CSV o una base de datos, y limpie un conjunto de datos determinado eliminando la(s) columna(s) menos importante(s), añadiendo columnas y limpiando los datos para los datos de texto estructurado (manipulación de datos).
LO-3.1.3	K2	Resumir varios pasos de preprocesamiento de datos para datos no estructurados (imágenes).
LO-3.1.4	K2	Resumir varios pasos de preprocesamiento de datos para datos no estructurados (texto).
HO-3.1.4	H1	Realizar pasos de preprocesamiento de datos en datos no estructurados (texto).
LO-3.1.5	K3	Aplicar varios métodos de imputación de datos a diferentes tipos de problemas.
HO-3.1.5	H3	Rellenar los valores de datos que faltan para un conjunto de datos determinado utilizando la media, el modo y el método KNN.
LO-3.1.6	K1	Enumerar varios tipos de gráficos (uni, bi y multivariante) utilizados para la visualización de datos.
LO-3.1.7	K2	Explicar el concepto de valores atípicos (outliers) y varias razones para su presencia.
LO-3.1.8	K3	Aplicar el método de trazado visual de cuadros para determinar los valores atípicos de un conjunto de datos

		determinado.
HO-3.1.8	H2	Dibuje un diagrama de caja para el conjunto de datos dado para identificar los valores atípicos.
LO-3.1.9	K3	Aplicar técnicas de reducción de la dimensionalidad - eliminación de características irrelevantes, análisis de componentes principales (principal component analysis, PCA).
HO-3.1.9	H3	Realizar la eliminación de características irrelevantes, PCA, para la reducción de la dimensionalidad en un conjunto de datos dados.
LO-3.2.1	K2	Explicar el papel de las métricas en los sistemas de ML.
LO-3.2.2	K1	Enumerar varios parámetros de evaluación de modelos para el aprendizaje supervisado y no supervisado.
LO-3.2.3	K2	Comparar la inercia y la puntuación de Rand ajustada como métricas para la agrupación no supervisada.
HO-3.2.3	H3	Aplicar inercia — suma de cuadrados dentro del grupo (within-cluster-sum-of-squares, WCSS) — y usar el método del codo para determinar el número óptimo de grupos.
LO-3.2.4	K2	Compare las métricas de apoyo, confianza y elevación para la minería de reglas de asociación no supervisada.
HO-3.2.4	H3	Calcular las métricas de apoyo, confianza y elevación para la minería de reglas de asociación no supervisada.
LO-3.2.5	K2	Explicar la matriz de confusión.
LO-3.2.6	K3	Aplicar la fórmula para calcular las métricas de exactitud, precisión, memoria, especificidad y puntuación F1 para la clasificación supervisada y compararlas.
HO-3.2.6	H3	Calcular la exactitud, la precisión, la memoria, la especificidad y la puntuación F1 para la clasificación supervisada.
LO-3.2.7	K3	Aplicar la fórmula para calcular varias métricas (error cuadrático medio (RMSE) y R-cuadrado) para la regresión supervisada.
HO-3.2.7	H2	Calcular los errores RMSE y R-cuadrado para la regresión supervisada.

LO-3.3.1	K1	Definir la necesidad de validar los modelos dividiendo el conjunto de datos disponible en conjuntos de datos de entrenamiento, validación y prueba.
LO-3.3.2	K2	Resumir el concepto de subajuste y sobreajuste, y de compensación de varianza de compensación de varianza de sesgo (bias-variance tradeoff).
HO-3.3.2	H0	Demostrar el subajuste y el sobreajuste para mostrar la compensación de la varianza de sesgo.
LO-3.3.3	K3	Aplicar los métodos de prueba dividida, validación cruzada de K (K-fold Cross-Validation), bootstrap y métodos de validación cruzada dejando uno fuera (leave-one-out cross-validation methods).
HO-3.3.3	H1	Aplicar el método de validación cruzada de K (K-fold Cross-Validation) en un algoritmo y comparar los distintos resultados.
LO-3.4.1	K1	Recordar las características de los cuatro tipos de análisis — descriptivo, exploratorio, predictivo, prescriptivo — y su uso en el ML

3.1 Preparación y preprocesamiento de datos

3.1.1 Pasos de la preparación de datos y del preprocesamiento

LO-3.1.1	K1	Recordar los pasos necesarios para la preparación y el preprocesamiento de los datos y la necesidad de una limpieza completa de los datos.

Los datos de entrada pueden ser en forma de tablas de base de datos, archivos CSV (valores separados por comas), o pueden ser datos no estructurados tales como imágenes, audios, videos o textos en ejecución. Los datos necesarios se obtienen de diversas fuentes, internas y externas.
Después de la adquisición, los datos necesitan ser limpiados a fondo y procesados antes de que puedan ser alimentados a los algoritmos para su entrenamiento y prueba.
Para la preparación y el preprocesamiento de los datos se realizan los siguientes pasos:

- Manipulación de datos

- Filtrado de datos
- Las actividades de preprocesamiento de datos incluyen
    - Imputación de datos, es decir, tratamiento de los valores que faltan
    - Visualización de datos para obtener una visión general y tratar anomalías y valores atípicos
    - Análisis de correlación y reducción de dimensiones

Es necesario realizar varios pasos específicos de preprocesamiento de formato de datos (por ejemplo, imagen, texto) para que el formato de datos sea adecuado para el entrenamiento. Cuando el volumen de datos es demasiado grande, reduzca los datos sin perder la información. En caso de que falten datos estructurados, es posible que sea necesario rellenar los valores de los datos. Todos estos pasos de preprocesamiento de datos son necesarios para obtener la precisión deseada y una mejor predictibilidad en los modelos.

3.1.2 Preparación de datos

LO-3.1.2	K3	Aplicar la lectura y manipulación de datos, incluyendo los pasos de filtrado para los datos estructurados.
HO-3.1.2	H1	Utilizar el código (lenguaje de su elección) para leer datos de varias fuentes de datos tales como un archivo Excel, un archivo CSV o una base de datos y limpiar un conjunto de datos dado, eliminando la(s) columna(s) menos importante(s), añadiendo columnas y limpiando los datos para datos de texto estructurado (manipulación de datos).

La preparación de datos incluye:

1. **Manipulación de datos**: Cambiar la estructura de los datos dados, por ejemplo, añadir una nueva columna, eliminar algunas líneas, etc.
2. **Filtrado de datos**: Reducir el tamaño de los datos estructurados (tabla/matriz) y no estructurados (imagen/texto) para mejorar la calidad de los datos.

3.1.3 Procesamiento de datos no estructurados (imágenes)

LO-3.1.3	K2	Resumir varios pasos de preprocesamiento de datos para datos no estructurados (imágenes).

Eliminar el ruido de la imagen y redimensionarla son las operaciones habituales que se realizan en las imágenes para el diseño de los algoritmos de visión por computador. [UDI]

3.1.4 Tratamiento de datos no estructurados (texto)

LO-3.1.4	K2	Resumir varios pasos de preprocesamiento de datos para datos no estructurados (texto).
HO-3.1.4	H1	Realizar pasos de preprocesamiento de datos en datos no estructurados (texto).

El preprocesamiento de datos de texto puede realizarse en múltiples pasos de cambios sintácticos, dependiendo de la necesidad del modelo ML. Por ejemplo, la eliminación de números, la conversión de mayúsculas a minúsculas, la eliminación de signos de puntuación, de los espacios en blanco, la eliminación de palabras de parada, la ejecución de la derivación o la lematización, etc. [UDT]

3.1.5 Imputación de datos

LO-3.1.5	K3	Aplicar varios métodos de imputación de datos a diferentes tipos de problemas.
HO-3.1.5	H3	Rellenar los valores de datos que faltan para un conjunto de datos determinado utilizando la media, el modo y el método KNN.

Los datos recogidos en campo pueden tener valores nulos o faltantes, por lo que es necesario sustituir los valores nulos por algunos valores apropiados. Los valores nulos o faltantes pueden ser imputados con medidas de tendencia central (media, mediana o modo), método K-vecino más cercano (K-Nearest-Neighbor) [DI1], o un enfoque basado en regresión.

3.1.6 Visualización de datos

LO-3.1.6	K1	Enumerar varios tipos de gráficos (uni, bi y multivariante) utilizados para la visualización de datos.

Visualizar los datos ayuda a comprender su estructura y la relación entre sus atributos, lo que no es posible con sólo mirar los números o el texto proporcionado. Hay varios tipos de visualizaciones. Los métodos de visualización más utilizados son: gráficos de líneas para valores continuos,

histogramas para valores discretos, gráficos de recuadros, gráficos de barras, gráficos circulares, etc. Ofrecen una visión significativa de los datos disponibles desde el principio.
Tipos de gráficos:
Univariable: La forma más simple de análisis, en la que los datos analizados son una sola variable. Por ejemplo, la edad de una población o el peso de una población, etc. Se analizan individualmente y su relación nunca se considera. Para el análisis de los datos univariados, se utilizan gráficos de líneas, histogramas, distribución de frecuencias, gráficos de barras y gráficos de bigotes (box plots).

Bivariable: Este tipo de análisis se lleva a cabo para encontrar la relación entre dos variables en el conjunto de datos dado. Trazar una variable contra otra en un plano XY ayuda a encontrar la relación de primera mano entre dos variables. Por ejemplo, la relación entre la edad y el peso de la población en cuestión. Para este tipo de análisis, puede utilizar gráficos de dispersión o correlogramas.

Multivariante: El análisis de tres o más variables. Los gráficos de malla y los gráficos 3D son algunas de las formas en que se pueden visualizar datos multivariados y descubrir relaciones entre ellos.

3.1.7 Detección de anomalías y de valores atípicos

LO-3.1.7	K2	Explicar el concepto de valores atípicos y las diversas razones de la presencia de valores atípicos.

Las observaciones que no siguen el patrón esperado para un conjunto de datos determinado entran en la categoría de valores atípicos, por ejemplo, la detección de fraudes o los ataques de hackers.
Si los valores atípicos no son frecuentes y no contribuyen en los eventos críticos, entonces pueden ser eliminados. Pero en la práctica, deben ser investigados a fondo antes de recortarlos del conjunto de datos.

Causas de los valores atípicos

- Error: Los valores atípicos en este caso son el resultado de un error en la medición, entrada de datos y muestreo, por ejemplo, los datos de temperatura registrados en grados Celsius para la mayoría de los registros, pero para algunos otros, en grados Fahrenheit, por error.

- Natural: Algunos valores atípicos pueden ocurrir en una situación natural, por ejemplo, si un incidente de inundación ocurre una vez cada 100 años, es un valor atípico natural.
- Intencional: Valores atípicos ficticios hechos para validar métodos de detección, por ejemplo, registros artificiales utilizados para probar casos extremos (corner scenarios).

3.1.8 Técnicas de detección de valores atípicos

LO-3.1.8	K3	Aplicar el método de diagrama de cajas y bigotes (box plot) para determinar valores atípicos a partir de un conjunto de datos determinado.
HO-3.1.8	H2	Dibuje un diagrama de caja y bigotes para el conjunto de datos dado para identificar los valores atípicos.

Mientras se calculan varias estadísticas con el conjunto de datos dado, a menudo es útil hacer uso de un diagrama de caja para ver la distribución de los datos. Los gráficos de caja ayudan a determinar la posición atípica del conjunto de datos dado.

Los extremos de la caja son los cuartiles superior e inferior. Los bigotes son las dos líneas fuera de la caja que se extienden hasta los umbrales de datos más altos y más bajos, más allá de los cuales los puntos de datos se consideran valores atípicos.

3.1.9 Reducción de la dimensionalidad

LO-3.1.9	K3	Aplicar técnicas de reducción de la dimensionalidad — eliminación de características irrelevantes, análisis de componentes principales (principal component análisis, PCA).
HO-3.1.9	H3	Realizar la eliminación de características irrelevantes, PCA, para la reducción de la dimensionalidad en un conjunto de datos dados.

Los problemas de ML a menudo tienen un gran número de características de entrada, pero no todas ellas contribuyen a la clasificación o a la salida de regresión. Cuanto mayor sea el número de funciones, más difícil será visualizar

el conjunto de entrenamiento. La técnica para reducir el número de variables bajo consideración se llama reducción de la dimensionalidad.
La necesidad de trabajar con menos dimensiones que las dimensiones originales surge de:

- Factores de coste y velocidad
- Requisito de memoria
- Evitar la redundancia
- Identificación de la parte más relevante de los datos para su procesamiento posterior

Métodos de reducción de la dimensionalidad:

- La eliminación de características irrelevantes es la eliminación de columnas que no contribuyen a la variable de salida:
    - El análisis univariado ayuda a eliminar las columnas cuyo valor no cambia en las filas. Por ejemplo, considere los datos de la transacción de ventas de una sola tienda minorista, luego las columnas como el nombre de la tienda, la ubicación de la tienda no cambiarán entre las filas y deben ser eliminadas.

    - Los datos de baja densidad pueden ser fácilmente eliminados después de la investigación para ahorrar espacio. Por ejemplo, las columnas basadas en el ID de la fila de la base de datos tienen un valor único para cada fila y deben ser eliminadas.

- El análisis bivariado elimina uno de los dos atributos de entrada altamente correlacionados (por lo tanto, también se conoce como análisis de correlación), por ejemplo, en los datos de inventario de una tienda, "precio de artículo" y "cantidad de artículo" son atributos altamente correlacionados.

- Análisis de componentes principales: PCA (Principal component analysis) reduce las dimensiones de los conjuntos de datos más grandes de forma extensiva, pero conserva la información al máximo. Deduce un nuevo conjunto de variables independientes (llamadas componentes principales) y las pone en orden decreciente de importancia. El número requerido de componentes principales superiores (por lo tanto, un número reducido de variables o dimensiones) puede entonces ser seleccionado, preservando la máxima información posible del conjunto de datos original.

3.2 Métricas

3.2.1 Papel de las métricas

LO-3.2.1	K2	Explicar el papel de las métricas en los sistemas de ML.

Las métricas son parámetros de evaluación para un modelo entrenado y pueden ser consideradas como la medida en que el modelo entrenado entrega resultados precisos y confiables. Para un tipo dado de algoritmo, las métricas pueden ser usadas para comparar modelos entrenados entre sí.

3.2.2 Métricas para el aprendizaje supervisado y no supervisado

LO-3.2.2	K1	Enumerar varios parámetros de evaluación de modelos para el aprendizaje supervisado y no supervisado.

Los objetivos del problema para los modelos de aprendizaje supervisado y no supervisado son diferentes. Por lo tanto, las métricas para evaluar los modelos son diferentes.

Tipo de aprendizaje	**Tipo de modelo**	**Métricas utilizadas**
No supervisado	Agrupamiento (Clustering)	• Inercia • Puntuación Rand ajustada
	Asociación	• Soporte • Confianza • Levantamiento
	Clasificación	• Exactitud • Precisión • Memoria/Sensibilidad • Especificidad • Puntuación F1

Supervisado	Regresión	• Error cuadrático medio (Root-mean-square-error, RMSE) • Error R-cuadrado (R-square error)

3.2.3 Inercia y Puntuación de Rand Ajustada

LO-3.2.3	K2	Comparar la inercia y la puntuación de Rand ajustada como métricas para la agrupación no supervisada.
HO-3.2.3	H3	Comparar la inercia y la puntuación de Rand ajustada como métricas para la agrupación no supervisada. Aplicar inercia — suma de los cuadrados dentro de cada grupo (WCSS) — y usar el método del codo para determinar el número óptimo de agrupamientos.

Para un modelo basado en agrupamientos no supervisado, la inercia o WCSS (suma de los cuadrados dentro de cada grupo) es la dispersión media de un grupo a través de todos los grupos descubiertos. El menor valor de inercia significa un mejor agrupamiento, ya que significa que los puntos de datos dentro de un agrupamiento están más cerca unos de otros. El tamaño del agrupamiento (y, por lo tanto, el valor de la inercia) disminuirá naturalmente a medida que el número de agrupamientos aumente. Sin embargo, la inercia deja de disminuir significativamente más allá de un cierto número de agrupamientos; este punto muestra el valor óptimo para la inercia y el número de agrupamientos para un conjunto de datos determinado; este método se conoce como el método del codo.

Cuando los valores reales de las etiquetas están disponibles para cada punto de datos, se prefiere la puntuación de Rand ajustada (**adjusted Rand score)** a la Inercia. Es una medida de la similitud entre las asignaciones de agrupamientos

(por el modelo) y las clases separadas reales

3.2.4 Métricas de Soporte, Confianza y Levantamiento

LO-3.2.4	K3	Aplicar la fórmula para calcular las métricas de apoyo, confianza y levantamiento para la minería de reglas de

		asociación no supervisada.
HO-3.2.4	H3	Calcular métricas de apoyo, confianza y levantamiento para la minería de reglas de asociación no supervisada.

El soporte El soporte para un conjunto de elementos mide la frecuencia con la que aparece en las operaciones. Por ejemplo, si el elemento 'pan' está presente en 7 de cada 10 transacciones totales en una tienda minorista, su aporte es del 70%.

La confianza mide la probabilidad de que aparezca el conjunto de elementos Y, dado que ha aparecido X.

El levantamiento se utiliza para eliminar escenarios en los que dos conjuntos de elementos ocurren juntos con mucha frecuencia (por lo tanto, el valor de la métrica de confianza será alto). Sin embargo, los dos conjuntos de elementos pueden no tener ninguna interdependencia. Además, esta métrica puede revelar si más ocurrencia del conjunto de elementos X significa más o menos ocurrencia del conjunto de elementos Y (es decir, asociación positiva o negativa entre X e Y).

3.2.5 Matriz de confusión

LO-3.2.5	K2	Explicar la matriz de confusión.

Las métricas de clasificación supervisadas se calculan utilizando una matriz de confusión compuesta por los recuentos de los verdaderos positivos, los falsos positivos, los verdaderos negativos y los falsos negativos

Matriz de confusión	Meta Positiva	Meta Negativa
Modelo positivo	Verdadero positivo (True positive, TP)	Falso positivo (False positive, TP)
Modelo Negativo	Falso negativo (False negative, FN)	Verdadero negativo (True Negative, TN)

3.2.6 Exactitud, Precisión, Recuperación, Especificidad y Puntuación de F1

LO-3.2.6	K3	Aplicar la fórmula para calcular las métricas de exactitud,

		precisión, recuperación, especificidad y puntuación F1 para la clasificación supervisada y compararlas.
HO-3.2.6	H3	Calcule la exactitud, la precisión, la recuperación, la especificidad y la puntuación F1 para la clasificación supervisada.

La **Exactitud** de un modelo muestra qué porcentaje o fracción de las clasificaciones totales fueron realizadas con exactitud por el modelo entrenado en un conjunto de datos de prueba. Esta métrica se convierte en una mala elección de métrica si una clase de datos domina sobre las demás.
Exactitud = (TP + TN) / (TP +TN + FP +FN).

La **Precisión** mide la precisión con la que el modelo clasifica los verdaderos positivos.
Precisión = TP/ (TP+FP).

La **Recuperación** mide hasta qué punto el modelo falló o no detectó los positivos.
Recuperación = TP/ (TP+FN).

Para minimizar los falsos positivos, se requiere una alta precisión, mientras que, para minimizar los falsos negativos, la recuperación debe ser alta.

Al igual que la precisión, pero opuesto a la recuperación, la **especificidad** mide la precisión con la que el modelo clasifica los verdaderos negativos.
Especificidad = TN / (TN + FP)

La **puntuación F1** se calcula como la media armónica de precisión y recuperación. Una puntuación baja de F1 representa la mala calidad del modelo en la detección de positivos. La puntuación F1 tendrá un valor entre 0 y 1. Cerca de 1 significa que hay buena calidad y que no hay datos falsos que perturben el resultado.

3.2.7 RMSE y R-cuadrado

LO-3.2.7	K3	Aplicar la fórmula para calcular varias métricas (error cuadrático medio (RMSE) y R-cuadrado) para la regresión supervisada.
HO-3.2.7	H2	Calcular los errores RMSE y R-cuadrado para la regresión

		supervisada.

Las métricas del modelo de regresión supervisada representan qué tan bien se ajusta la línea de regresión a los puntos de datos reales.

RMSE (error cuadrático medio — root-mean-square-error) es una medida de cuán lejos están los puntos de datos de la línea de regresión. Se mide como la desviación estándar de los errores de predicción. El valor de RMSE cambia si el mismo conjunto de datos se mide en una unidad diferente.

R-cuadrado es una medida de cuán mejor son las predicciones de la línea de regresión comparadas con el uso de la media como predictor. Su valor oscila entre 0 y 1 y es independiente de la unidad utilizada para medir los puntos de datos.

3.3 Evaluación del modelo

Los valores de las métricas dependen de cómo se eligen los puntos de datos para la formación y la validación. Por lo tanto, el aspecto clave para la evaluación de modelos es que los puntos de datos para la formación y la validación se seleccionen de forma totalmente imparcial

3.3.1 Conjuntos de datos de entrenamiento, validación y pruebas

LO-3.3.1	K1	Definir la necesidad de validar los modelos dividiendo el conjunto de datos disponibles en conjuntos de datos de entrenamiento, validación y pruebas.

El conjunto de datos de entrenamiento contiene los datos sobre los que se ha formado al modelo. El conjunto de datos de validación es utilizado por el algoritmo de aprendizaje de la máquina para evaluar si el entrenamiento fue efectivo. En cada ejecución de ML (que es un proceso de muchas iteraciones), el conjunto de datos de entrenamiento y el conjunto de datos de validación se combinan de nuevo y se dividen de forma diferente, de modo que el algoritmo utiliza diferentes combinaciones de datos de las que aprender.

El conjunto de datos de prueba es un conjunto de datos separado que se utiliza una vez finalizado el proceso de ML para validar si el algoritmo ha sido adecuadamente entrenado. El conjunto de datos de la prueba no debe utilizarse durante el proceso de entrenamiento. [Wiki1]

En la fase posterior a la formación, un modelo de ML se evalúa y prueba con un conjunto de datos diferente del conjunto de datos de entrenamiento. Sin embargo, el conjunto de datos para las fases de formación, validación y prueba debe provenir de la misma fuente o de fuentes similares para garantizar la eficacia de las métricas. Un modelo entrenado usando un tipo de conjunto de datos puede funcionar muy mal en un conjunto de datos originado de fuentes muy diferentes.iki1]

3.3.2 Subajuste y sobreajuste

LO-3.3.2	K2	Summarize the concept of underfitting and overfitting and bias-variance tradeoff.
HO-3.3.2	H0	Resumir los conceptos de subajuste (underfitting) y sobreajuste (overfitting) y compensación de la varianza de sesgo (bias-variance tradeoff).

Si un modelo de ML supervisado es demasiado simplista para encajar en los puntos de datos de entrenamiento (es decir, no representa la tendencia de los datos), es un ejemplo de subajuste. Por el contrario, un modelo de sobreajuste intenta ajustar demasiado los puntos de datos de la formación, lo que a menudo da lugar a una precisión de predicción deficiente durante las fases de validación o prueba posteriores.

La naturaleza de subajuste y sobreajuste de un modelo también puede explicarse en términos de sesgos y errores de varianza. Si un modelo es demasiado simplificado y no aprende de todas las características proporcionadas para representarlo, se dice que tiene un alto sesgo y sufre de una pobre precisión de predicción.

Si el rendimiento de la predicción del modelo varía mucho al cambiar ligeramente el conjunto de datos de entrenamiento, se dice que es un modelo de alta varianza (demasiado dependiente del conjunto de datos de entrenamiento).

Un buen modelo tiene que lograr un sesgo bajo y una varianza baja. Esto se conoce como la compensación de la varianza de sesgo.

3.3.3 Métodos de validación cruzada

LO-3.3.3	K2	Explique los métodos de validación cruzada dividida, K-fold, bootstrap y dejando uno fuera (leave-one-out)
HO-3.3.3	H1	Aplicar el método de validación cruzada K-fold en un algoritmo y comparar varios resultados.

La forma en que el conjunto de datos disponible se divide en conjuntos de datos de entrenamiento y validación puede dar lugar a un sesgo elevado o a una gran varianza. Para superar esto, se deben probar múltiples combinaciones de particiones antes de concluir las métricas del modelo. Algunos métodos útiles son la prueba dividida, el bootstrap, la validación cruzada K-fold y la validación cruzada dejando uno fuera (leave-one-out cross-validation). Cada uno de estos métodos repite el proceso de entrenamiento y validación varias veces y el rendimiento del modelo se promedia en todas las ejecuciones.

La **prueba dividida** divide los datos en partes para los conjuntos de datos de entrenamiento y de pruebas, pero en cada iteración en proporciones diferentes. Esto ayuda a revelar cómo las diferentes divisiones pueden producir diferentes resultados.

El **Bootstrap** funciona seleccionando puntos de datos aleatorios de todo el conjunto de datos para la formación y utiliza el conjunto de datos restante para su validación.

La **validación cruzada de K-fold** divide el conjunto de datos en partes k y utiliza los subconjuntos k-1 para la formación y el subconjunto restante para la validación.

La **validación cruzada dejando uno fuera** es similar a la validación cruzada K-fold, excepto que cada parte tiene un punto de datos, por lo que requiere más tiempo de ejecución.

3.4 Analítica

3.4.1 Tipos de análisis

LO-3.4.1	K1	Recordar las características de los cuatro tipos de análisis -descriptivo, exploratorio, predictivo, prescriptivo- y su uso en el ML.

El análisis es una de las tareas principales cuando se trata de comprender el conjunto de datos disponible. La analítica de datos se puede realizar en cuatro niveles y cada nivel subsiguiente es una extensión natural del nivel anterior.

El **análisis descriptivo** consiste en deducir el resumen estadístico de los datos en términos de medidas de tendencia central (media, mediana, modo), así como medidas de dispersión (varianza, desviación estándar). Esto ayuda a comprender el pasado y la respuesta: "¿Qué ha pasado?" [DA1]

La **analítica exploratoria** consiste en visualizar el conjunto de datos a un alto nivel para ver sus patrones y variaciones.

El **análisis predictivo** consiste en modelar las variables de entrada y predecir la probabilidad de los resultados.

La **analítica prescriptiva** es comparar todas las predicciones viables que resultan de la analítica predictiva y elegir/prescribir las mejores de ellas.

Capítulo 4 - Pruebas online de sistemas de AI

Palabras clave: Método de análisis lingüístico, chatbot

LO-4.1.1	K2	Explicar las partes de AI y no AI de una aplicación inteligente y sus necesidades de pruebas funcionales y no funcionales.
LO-4.1.2	K1	Mostrar las interacciones orientadas a la información y a la acción de partes de la AI y no AI.
LO-4.2.1	K3	Utilizar el método de diseño de pruebas de análisis lingüístico para generar escenarios de prueba.
HO-4.2.1	H1	Demostrar el uso del método de diseño de pruebas de análisis lingüístico para generar escenarios de prueba.
LO-4.3.1	K3	Hacer uso de los casos de prueba derivados de los requisitos y la arquitectura dados para probar un chatbot.
HO-4.3.1	H3	Probar un determinado chatbot a nivel de sistema y reportar errores.

4.1 Arquitectura de una aplicación de AI

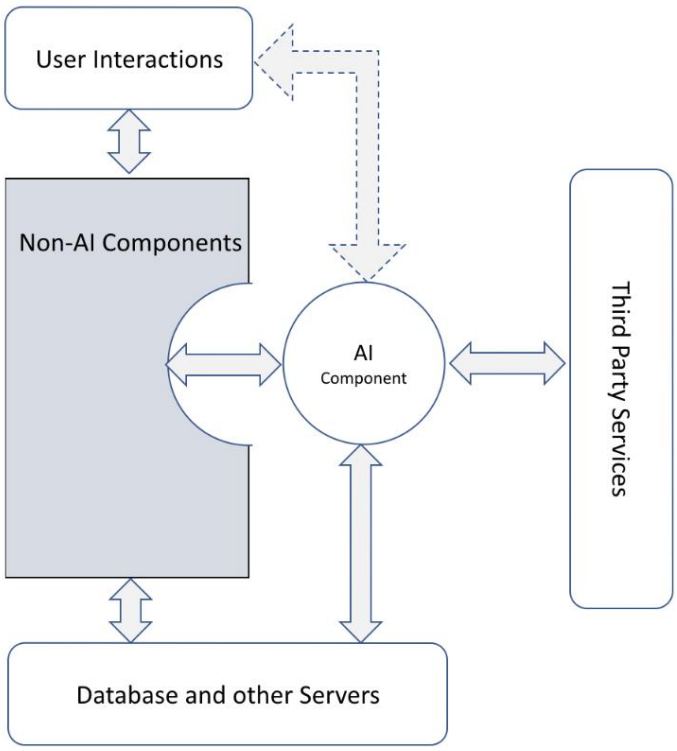
4.1.1 Componentes de una aplicación inteligente y sus necesidades de prueba

LO-4.1.1	K2	Explicar las partes AI y no AI de una aplicación inteligente y sus necesidades de pruebas funcionales y no funcionales.

Una aplicación de AI típica necesita ser examinada, ya sea una aplicación de AI monolítica, o un sistema general compuesto por un conjunto más pequeño de componentes de AI invocados desde una aplicación no AI más grande. Este análisis es necesario para entender cómo varios componentes, AI y no-AI, trabajan juntos para proporcionar la funcionalidad deseada. Las interfaces entre estos diversos componentes deben entenderse desde el punto de vista de:

- Datos de entrada recibidos
- Resultado generado

- Medidas adoptadas por los respectivos componentes
- Interacciones directas del usuario, si las hubiera
- Interacciones con sistemas de terceros, si las hubiera
- Frecuencia del entrenamiento
- Número de entrenamiento paralelos, si es posible
- Restricciones tales como
    - Tiempo
    - Duración
    - Rangos de entradas y salidas
- Condiciones previas necesarias
- Supuestos/configuraciones comunes
- Visualización de la cadena de entradas y salidas con las transformaciones relevantes, si las hubiera
- Error y excepciones y su tratamiento por parte de
    - El componente
    - Cadena de componentes y manejo final
- Registro (Logging)



Además, es necesario que la razón por la que el sistema llega a la solución dada sea explicable. Además, es necesario identificar escenarios de prueba para

estos sistemas de AI utilizando técnicas como el análisis lingüístico y las pruebas exploratorias.

La otra cuestión de complejidad que debe garantizarse es la presencia o ausencia de combinaciones de sistemas de AI. Los sistemas de AI independientes pueden ser insuficientes para especificar completamente un problema del mundo real. Para manejar estos escenarios, se utilizan combinaciones de sistemas de AI para modelar el problema. Para ello, se requiere una estrategia de prueba para manejar combinaciones de sistemas de AI.

En la interacción AI - no AI, es necesario garantizar la cobertura de las pruebas del componente AI, el componente no AI y la interacción que implica el traspaso entre las dos partes.

A continuación, se ofrece un ejemplo de prueba de un sistema de este tipo:
En una aplicación de correo electrónico, el autocompletado de frases es proporcionada por un componente de AI. La interfaz de usuario del sistema de correo electrónico representa la parte no AI y el componente AI interactúa con ella para proporcionar el texto sugerido. La parte no AI muestra la sugerencia y actúa sobre la entrada del usuario. Si la sugerencia es aceptada, esto podría resultar en el almacenamiento de algunos datos para el aprendizaje futuro.

El mismo sistema de correo electrónico también proporciona corrección ortográfica online, ya sea a través de un componente AI o no.

Si analizamos las interfaces, encontramos que la interfaz de usuario (componente no AI) está pasando el texto parcialmente escrito al componente AI. Si hay una sugerencia del componente de AI, el texto se muestra en la interfaz de usuario. Las limitaciones relacionadas con el tiempo y la duración son que la sugerencia tiene que hacerse en un plazo de unos pocos cientos de milisegundos a partir de que se introduce el texto y esto tiene que ser un proceso continuo. El probador necesita determinar si el sistema está respondiendo lo suficientemente rápido. Si hay problemas de tiempo y duración, como que las predicciones del modelo sean lentas, para cuando el sistema

muestre la oración sugerida, el usuario puede haber escrito texto adicional. Esto puede hacer que la frase sugerida sea incorrecta gramaticalmente o la sugerencia completamente incorrecta.

Un ejemplo de problemas en la interacción entre las partes de la interfaz gráfica de usuario (GUI) y las partes de una salida que no son de la interfaz gráfica de usuario (AI) se puede relacionar con la visualización del texto sugerido. El texto sugerido puede aparecer en un lugar incorrecto de la interfaz gráfica de usuario. La frase sugerida debe diferenciarse visualmente del texto real que se está escribiendo y necesita cambiar, basándose en las acciones del usuario. Cualquier error en estos comportamientos también caería bajo las interacciones GUI y no GUI (AI).

Un ejemplo de la parte de manejo de errores/excepciones es el escenario donde el componente de terminación de oraciones falla (por alguna razón) y la GUI es capaz de manejarlo.

La configuración del sistema de correo electrónico incluye la configuración de idioma y la configuración del diccionario. Por ejemplo, el inglés como idioma y el inglés de EE.UU. como configuración del diccionario. El sistema de finalización de oraciones debería proporcionar sugerencias en inglés estadounidense. En este caso, el sistema de corrección ortográfica y gramatical debería utilizar también el inglés estadounidense. Esto significa que las frases formadas por el componente de AI no deberían ser marcadas como texto incorrecto por el componente de corrección ortográfica debido a un desajuste en la suposición/ajuste de los dos componentes interactivos de AI o AI-no AI.

Las imágenes siguientes muestran algunos ejemplos de problemas descubiertos en un sistema de este tipo, A y B muestran la marcación de un nombre como un error ortográfico y no el otro, mientras que ambos nombres forman parte de la libreta de direcciones.

Por otro lado, B y C muestran la diferencia de velocidad y cacheo. Cuando se escribe por primera vez, se completa el nombre, pero el nombre no se marca como un error durante unos segundos y luego se marca como un error. Sin embargo, una vez marcado como un error, incluso la eliminación del nombre y

la posterior autocompletación lo marca como un error inmediatamente.

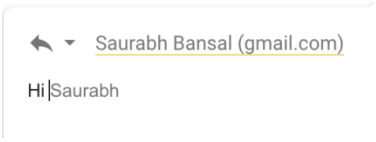


Imagen A

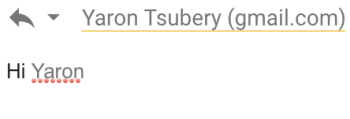


Imagen B

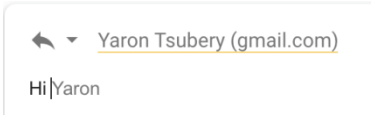


Imagen C

4.1.2 Interacción de partes AI y no AI

LO-4.1.2	K1	Mostrar la interacción orientada a la información y a la acción de partes de AI y no AI.

En las llamadas orientadas a la información, la API o el servicio de la aplicación AI se invoca desde una aplicación final que puede ser no AI. Típicamente, el componente AI es invocado para una respuesta. Por ejemplo, un algoritmo de detección de fraude funciona con la idea de llamar a un modelo predefinido entrenado para que devuelva si una transacción de entrada es fraudulenta o no. Algunas de las pruebas importantes para probar las interacciones son relativas a:

- Pruebas basadas en valores límite - tanto de entrada como de salida

Casos de prueba inusuales (también conocidos como casos de prueba esquina - Unusual test cases (a.k.a. Corner test cases)

- Pruebas relacionadas con el tamaño y el tipo de datos que deben pasar
- Pruebas de manejo de excepciones
    - No se recibió respuesta
    - Se rompió el circuito de retroalimentación de la solicitud-respuesta para las acciones
- Pruebas de datos de entrada erróneas
- Pruebas de datos de salida erróneas
- La solicitud no pudo ser completada
- Respuesta no recibida
- Pruebas de rendimiento
- Pruebas de seguridad
- Pruebas de robustez

Las interacciones entre los componentes de AI y no AI pueden tener un impacto en la experiencia del usuario. Las interacciones pueden ser de los siguientes tipos:

- Señalar solamente (sólo para alimentar el sistema con información — Flag only)
- Orientado a la acción, por ejemplo, generar una respuesta para el usuario, O clasificar el problema
- Escenarios de fallo de APIs interactivas en los que las APIs no devuelven un resultado
- Entrega de componentes AI a componentes no AI y viceversa

Cuando se diseñan pruebas para los sistemas de AI, éstas son de los siguientes niveles de prueba:

- Pruebas que sólo prueban la parte de la AI
- Pruebas que sólo prueban la parte que no es de AI
- Pruebas para la integración de ambas partes
- Respuesta en caché vs. respuestas aprendidas

En un sistema no autónomo más extenso, es necesario considerar la cobertura del modelo de AI desplegado, así como la gestión del rendimiento del modelo de AI desplegado. Después del desarrollo del componente AI, el sistema necesita ser probado en un entorno de despliegue.

4.2 Análisis lingüístico Método de diseño de la prueba

4.2.1 Diseño de pruebas basadas en el análisis lingüístico

LO-4.2.1	K3	Utilizar el método de diseño de pruebas de análisis lingüístico para generar escenarios de prueba.

El análisis lingüístico se utiliza para diseñar un gran número de escenarios, incluso cuando los requisitos están mal documentados [LA1]. Un análisis lingüístico de los requisitos identifica los objetos de prueba y las medidas que deben tomarse al respecto. Este método ayuda a encontrar casos de prueba esquina (inusuales), un requisito clave para probar sistemas de AI.

El método funciona como se indica a continuación:

1. Identificar los sustantivos que representan los objetos de prueba y los verbos que representan las acciones.
2. Identificar las propiedades del sustantivo.
3. Identificar las propiedades de las propiedades y repetirlas hasta que no se puedan descubrir más propiedades o hasta que se considere que se ha alcanzado la profundidad suficiente.
4. Identificar adverbios y adjetivos aplicables a los sustantivos y verbos para identificar más propiedades.
5. Usar 5W1H (Qué, Por qué, Dónde, Cuándo, Cuál, Cómo) en los verbos para identificar las combinaciones sustantivo-verbo que dan las pruebas funcionales y no funcionales.
6. Hacer las siguientes preguntas en cada escenario:
    a. ¿Quién/qué es el agente?
    b. ¿Cuáles son los instrumentos o medios para lograr el escenario?
    c. ¿Cuál es el propósito de la acción?
    d. ¿Cuál es la trayectoria de la acción?
    e. ¿Cuál es el origen de la acción?
    f. ¿Cuál es el motivo de la acción?
    g. ¿Quién posee los medios y los resultados de la acción?
    h. ¿Dónde tiene lugar la acción?
    i. ¿Quién es el receptor de la acción?

Estas preguntas siguen las reglas gramaticales del Sánscrito [LA2].
En el caso de probar sistemas de AI, los datos son el caso de prueba. El método dado permite la identificación tanto de los datos como de la combinación de acciones (escenarios).

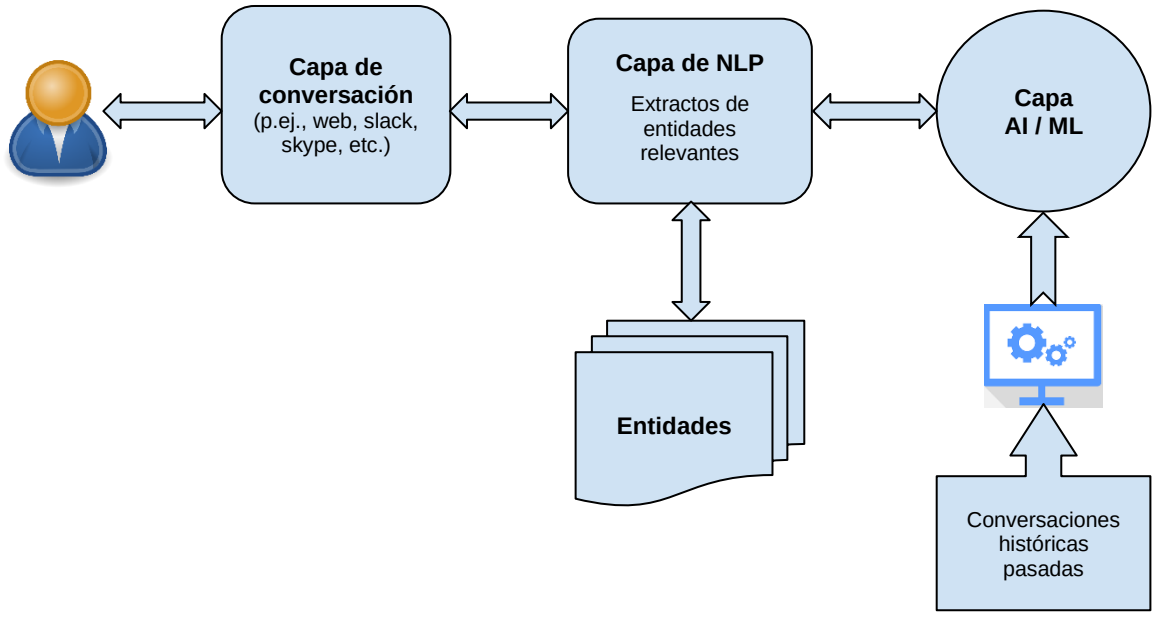
4.3 Pruebas de sistemas de inteligencia artificial

4.3.1 Prueba de un Chatbot

LO-4.3.1	K3	Hacer uso de los casos de prueba derivados de requisitos y arquitectura dados para probar un chatbot
HO-4.3.1	H3	Probar un chatbot dado a nivel de sistema y reportar errores.

Los Chatbots se utilizan hoy en día en varios contextos, que van desde los bots de servicio al cliente, los bots de información, hasta los bots de apoyo.

La arquitectura lógica muestra las capas clave de un chatbot, incluyendo la capa de conversación, la capa NLP (Procesamiento del lenguaje natural, Natural Language Processing), la capa AI/ML y la capa de conectividad. La capa de conversación gestiona la interacción del interfaz frontal (front-end) con el usuario final, por ejemplo, Skype, Slack, Facebook messenger o un interfaz frontal (front-end) web. La capa de NLP extrae las entidades e información relevantes de los enunciados del usuario final. La capa AI/ML decodifica la intención real del enunciado, habiendo sido entrenada en conversaciones históricas pasadas alimentadas como datos para el sistema.



Los casos de prueba deben cubrir las variaciones de los datos de entrada en

cada capa individual.
La capa de conversación necesita ser probada para la transmisión de datos correcta desde el interfaz frontal (front-end) al interfaz motor (backend) y viceversa. En la capa de PNL, uno debe probar si la limpieza de datos y el preprocesamiento de los enunciados de texto en bruto está ocurriendo correctamente y si las entidades extraídas son suficientes para transmitir el contexto en una conversación en curso. La capa o modelo AI/ML (entrenado en conversaciones históricas) debe ser probado por su precisión para predecir la intención correcta basada en entidades extraídas (por la capa de NLP), por ejemplo, para un texto de entrada 'Hola' / 'Buen día' / 'Saludos', etc. La intención de 'Bienvenida' debe ser predicha.

Además, el chatbot debería ser capaz de saltar entre varios intentos sin problemas si la secuencia de flujo de conversación cambia bruscamente. Por lo tanto, deben identificarse los casos de prueba que cubren la conmutación dinámica de la intención.

Los requisitos no funcionales deben ser probados para el rendimiento de la interfaz frontal principalmente, si es capaz de soportar muchas invocaciones concurrentes de los clientes.

Capítulo 5 - Inteligencia Artificial Explicable

Palabras clave: interpretar/explicar el modelo ML, explicable, LIME (Explicaciones Modelo-Agnósticas de Interpretable Local, Local Interpretable Model-Agnostic Explanations), CAM (Fabricación asistida por computador, Computer-Aided Manufacturing)

LO-5.1.1	K2	Explicar la necesidad de la AI explicable (Explainable AI, XAI).
LO-5.1.2	K3	Utilizar LIME para explicar un modelo de AI.
HO-5.1.2	H1	Utilizar LIME en un clasificador de imágenes, así como en un clasificador de texto.
LO-5.1.3	K3	Utilizar Grad-CAM para explicar los modelos de CNN (Red neuronal convolucional, Convolutional neural network).
HO-5.1.3	H1	Utilizar Grad-CAM para hacer más transparente CNN.

5.1 AI explicable (Explainable AI, XAI)

5.1.1 La AI explicable y su necesidad

LO-5.1.1	K2	**Explicar la necesidad de la AI explicable (Explainable AI, XAI).**

Una vez que un modelo de ML es entrenado, debería funcionar con un nivel definido de precisión y para todas las variaciones definidas de los escenarios. Si la calidad de las predicciones del modelo es insuficiente para algunos escenarios y funciona a la perfección para otros, es probable que sea un modelo sesgado. Un conjunto de datos para el que la calidad del modelo es inferior al nivel de calidad definido representa un defecto.
Como probador, resulta muy difícil descubrir todos estos defectos sin utilizar un proceso sistemático. Se necesita examinar el comportamiento del modelo y su variación con el cambio en los factores de entrada para deducir una aproximación de la relación entre entrada y salida. Tal relación aproximada puede no ser una sustitución del modelo real, pero puede ser lo suficientemente buena como para revelar un posible sesgo. Una mirada al comportamiento del

modelo ayuda a evaluar su calidad general y la viabilidad del modelo para su despliegue. Entre otras razones por las que es necesario explicar o interpretar los modelos se encuentran las medidas de seguridad, la aceptación social, la detección de sesgos o la curiosidad humana y el aprendizaje sobre el modelo [EA1].

No todos los modelos se pueden explicar. Cuanto más complejo sea el modelo, menos probable será que lo interprete o lo explique. El resultado de los modelos no DL (Deep Learning, por ejemplo, bosque aleatorio [Wiki2], árboles de decisión [Wiki3], regresión lineal [Wiki4], etc., puede explicarse de forma conveniente en términos de las variables de entrada.

Los modelos de DL son intrínsecamente complejos en su implementación, por lo que es mejor examinar el modelo como una caja negra, es decir, observar las variaciones de los resultados por pequeñas perturbaciones de las variables de entrada y aproximarse al modelo subyacente mediante un modelo simple e interpretable.

Algunos de los algoritmos, herramientas y enfoques populares y fáciles de usar para la interpretación de modelos incluyen: Explicaciones Modelo-Agnósticas de Interpretable Local (Local Interpretable Model-agnostic Explanations, LIME) [EA2] y Fabricación asistida por computador (Computer-Aided Manufacturing Class, CAM) [EA3].

5.1.2 LIME

LO-5.1.2	K3	Utilizar LIME para explicar un modelo de AI.
HO-5.1.2	H1	Utilizar LIME en un clasificador de imágenes, así como en un clasificador de texto.

LIME es alimentado con una muestra para estudiar las predicciones del modelo y sus variaciones más cercanas, y revela las características de entrada responsables de la salida prevista del modelo. LIME genera suficientes variantes cercanas de la entrada de la muestra y obtiene el resultado para cada una de las variantes. Por lo tanto, intenta aproximarse a cómo las pequeñas

variaciones en la entrada modifican la variable de salida. Dado que todas las variantes generadas y estudiadas por LIME están en la proximidad de la muestra dada, las explicaciones de LIME se llaman “locales” (a la entrada de la muestra). Además, es un método agnóstico al modelo, ya que LIME no requiere que se mire el modelo o el algoritmo.

LIME puede utilizarse en un clasificador de imágenes para generar explicaciones. Deduce porciones de imagen que juegan un papel significativo en la determinación del resultado. Permite a un probador relacionarse y descartar que el modelo no esté basando sus deducciones en elementos irrelevantes.
Del mismo modo, para un clasificador de texto, LIME puede indicar las palabras (porciones de texto) que conducen a la categorización del texto de muestra en una de las clases predefinidas. Esto ayuda a un probador a evaluar si el modelo está basando sus deducciones en palabras irrelevantes.

5.1.3 CAM para redes neuronales (NN)

LO-5.1.3	K3	Utilizar Grad-CAM (Mapas de activación de clases ponderados por grados, Gradient-weighted Class Activation Mapping) para explicar los modelos de CNN.
HO-5.1.3	H1	Utilizar Grad-CAM para hacer más transparente CNN.

Las redes neuronales (NN) convolucionales CNNs son muy útiles, pero no son transparentes en cuanto a por qué llegaron a una conclusión en particular. Con el fin de aportar transparencia a estos modelos, el Mapeo de Activación de Clases (Gradient-based Class Activation Mapping, Grad-CAM) visualiza las regiones de entrada que son importantes para las predicciones de estos modelos. Utiliza la información de gradiente específica de la clase que fluye hacia la capa convolucional final de una CNN y produce un mapa de localización general de las regiones importantes en la imagen.

Capítulo 6 - Riesgos y estrategia de ensayo de los sistemas de AI

Palabras claves: modelo pre-entrenado, deriva conceptual

LO-6.1.1	K1	Recordar los riesgos de probar los sistemas de AI.
LO-6.1.2	K2	Explicar los problemas asociados con el uso de modelos pre-entrenados de terceros en los sistemas de producción.
LO-6.1.3	K2	Explicar la necesidad de probar el modelo entrenado nuevamente debido a la deriva del concepto.
LO-6.1.4	K1	Definir el entorno de pruebas para los sistemas de AI y recordar sus desafíos.
LO-6.2.1	K1	Recordar las estrategias de pruebas para aplicaciones de AI, especialmente teniendo en cuenta los tipos, fases y niveles de pruebas.

6.1 Riesgos en las pruebas de AI

6.1.1 Riesgos en las pruebas de sistemas de AI

LO-6.1.1	K1	Recordar los riesgos de probar los sistemas de Inteligencia Artificial.

La prueba de los sistemas de AI presenta algunos riesgos adicionales, además de los riesgos asociados con los sistemas que no son de AI. Algunos de estos riesgos son

- Pruebe los desafíos relacionados con los criterios de prueba
    - Con un gran número de pruebas (cada conjunto de datos es una prueba), ¿cómo se verifica?
        - la exactitud de las pruebas
        - la integridad de las pruebas
    - Falta de un oráculo de prueba fiable que indique cuál debe ser la salida correcta y la entrada arbitraria
    - Identificación de falsos positivos y falsos negativos. Los falsos positivos son aún más fáciles de identificar porque se deben investigar las fallas. Los falsos negativos son difíciles de identificar porque estas pruebas se muestran como pasadas
- Desafíos relacionados con los datos de prueba
    - En AI, los datos de prueba son iguales a un caso de prueba, sobre todo para pruebas offline. Lo ideal es que se requiera una gran cantidad de datos de prueba para realizar la prueba.
    - La disponibilidad de datos puede ser un problema

    - La calidad de los datos puede ser mala, lo que requiere una limpieza de datos
    - Se requieren datos etiquetados/anotados para el entrenamiento y las pruebas.
    - Los casos de esquina de los sistemas de ML son difíciles y costosos de generar
- Retos relacionados con los costes
    - Para aplicaciones complejas, el tipo de hardware requerido para entrenar y probar el modelo offline puede ser costoso
    - Los costos de energía de la capacitación DNN (red neuronal profunda) son generalmente muy altos
- Desafíos relacionados con las habilidades y las herramientas
    - Los requisitos de habilidades para probar sistemas de AI son altos. El probador necesita entender cómo se construyen los sistemas de AI y cómo deben ser probados
    - La falta de información estructurada, herramientas y marcos para la prueba de sistemas de AI
- Comprensión de dominios y desafíos relacionados con los sesgos
    - Las pruebas formales de la calidad óptima de un algoritmo no garantizan que una aplicación implemente o utilice el algoritmo correctamente
    - Se requiere una cobertura completa de los casos de entrada, basada en el dominio, para evitar sesgos, una cobertura incompleta y la posibilidad de accidentes.

6.1.2 Riesgo de utilizar modelos preentrenados

LO-6.1.2	K2	Explicar los problemas asociados con el uso de modelos pre-entrenados de terceros en los sistemas de producción.

Algunas organizaciones han puesto a disposición sus modelos preaprendidos y muchos desarrolladores de AI los utilizan.
Por ejemplo, modelos de ImageNet tales como Inception, VGG, AlexNet, etc.
Los modelos pueden tener sus propios sesgos y defectos que pueden necesitar ser descubiertos mediante pruebas. Debido a que estos son desconocidos o indocumentados, los sistemas construidos utilizando modelos pre-entrenados pueden fallar de manera imprevista o producir resultados que pueden ser sub-óptimos en algunas situaciones.

6.1.3 Riesgo de deriva del concepto (Concept Drift, CD)

LO-6.1.3	K2	Explicar la necesidad de probar el modelo entrenado nuevamente debido a la deriva del concepto.

Los modelos de trabajo pueden degradarse con el tiempo debido al cambio en la relación entre los elementos de entrada y salida. Por ejemplo, el impacto de los anuncios publicitarios y de otros tipos de campañas de marketing resulta en un cambio en el comportamiento de los clientes potenciales cada cierto tiempo. Esto se conoce como Deriva de concepto (Concept Drift, CD). Para averiguar la ocurrencia de la deriva del concepto, se requieren pruebas periódicas de los modelos de trabajo y de los modelos integrados con muestras de datos recientes. Para ello, el reentrenamiento del modelo y la repetición del ciclo de varias pruebas y validaciones se realiza en las fases offline e integrada. [JB2] Existen varias formas de manejar la deriva de concepto, tales como:

- No hacer nada
- Reentrenar el modelo con datos recientes
- Actualizar periódicamente el modelo utilizando los últimos datos del modelo actual
- Aprender el cambio — Un enfoque de conjunto en el que el modelo actual se deja sin cambios. Un nuevo modelo toma los resultados del modelo actual y aprende a corregir las predicciones
- Detectar y elegir el modelo — detectar la deriva del concepto, si es posible, y elegir un modelo apropiado.

6.1.4 Desafíos del entorno de prueba de la AI

LO-6.1.4	K1	Definir el entorno de pruebas para los sistemas de AI y recordar sus desafíos.

Los entornos de prueba de los sistemas de inteligencia artificial pueden ser muy complejos, debido a los diferentes casos de uso, contextos y diversas formas y pasos del preprocesamiento de datos. Desde el punto de vista de las pruebas offline, las necesidades del entorno son más exigentes que desde el punto de vista de las pruebas online. Se necesita un gran tamaño de almacenamiento de datos, un alto requerimiento de ancho de banda de red y también una mayor potencia computacional para entrenar/ejecutar el modelo.

6.2 Estrategia de la prueba

6.2.1 Estrategia de prueba para probar aplicaciones de AI

LO-6.2.1	K1	Recordar las estrategias de pruebas para aplicaciones de AI, sobre todo teniendo en cuenta los tipos, fases y niveles de las pruebas.

Una aplicación basada en la AI del mundo real puede emplear uno o más componentes de AI y no AI. La estrategia de prueba para un sistema de este tipo incluirá dimensiones de prueba convencionales, así como nuevos factores específicos para los componentes de AI y su integración con otros componentes del sistema.

Algunas de las consideraciones convencionales de la estrategia de prueba son:

- **Nivel de prueba requerido para el sistema** — prueba unitaria, prueba de integración, prueba de sistema, prueba de aceptación y prueba de integración de sistema como se describe en el programa de estudios de ISTQB® CTFL. [ISTQB-CTFL2018]
- **Técnicas de prueba** — Caja blanca, caja negra y caja gris
- **Pruebas funcionales y no funcionales** — sobre todo pruebas de seguridad, rendimiento, robustez y escalabilidad
- **Uso de la automatización de pruebas**

Para probar las aplicaciones basadas en la AI, se necesita tener en cuenta los siguientes aspectos adicionales:

- **Pruebas offline (funcionales)** — Probar el modelo de AI entrenado es un paso adicional que debe realizarse como parte del Ciclo de vida del desarrollo de software (Software Development Life Cycle, SDLC). También es necesario tener en cuenta las aptitudes requeridas para realizar las pruebas en esta fase.

- **Pruebas offline (no funcionales)** — Probar el modelo de AI entrenado para varios aspectos no funcionales. El modelo necesita ser probado en cuanto a la velocidad, utilización de recursos, concurrencia y carga, y escalabilidad antes de su despliegue.

- **Pruebas de caja negra** — La mayoría de los modelos de AI se exponen como APIs y se invocan como cajas negras en general. Un componente crítico que falta en los sistemas de AI es la falta de oráculo de prueba, lo que dificulta las pruebas de caja negra. Técnicas como las pruebas metamórficas se están volviendo populares para el ML ya que no se requiere un oráculo. Si los casos de prueba se ejecutan con éxito, se realizan variaciones en los casos de prueba y se examinan los resultados. La necesidad de que la salida sólo satisfaga las relaciones no requiere ningún oráculo. [Wiki5]

- **Pruebas de caja blanca** — En general, las pruebas de caja blanca para sistemas ML son difíciles ya que, excepto en algunos modelos simples, el funcionamiento interno del modelo no es claro ni accesible. Sin embargo, recientemente, han surgido algunas técnicas de caja blanca para los sistemas de ML. Un ejemplo es DeepXplore, un producto de pruebas automatizadas de caja blanca. Usando la cobertura neuronal, los investigadores fueron capaces de exponer miles de comportamientos únicos e incorrectos en los casos de esquina. [DX1]

- **Adquisición y preprocesamiento de datos** — Los datos disponibles pueden no estar siempre en condiciones de ser alimentados a un modelo para su entrenamiento o prueba. Esto ha sido discutido en detalle en 3.1 Preparación y preprocesamiento de datos.

- **Conversión de las implementaciones del entorno de desarrollo a la producción** — por ejemplo, convertir el portátil Jupyter en código Python que se ejecuta en el servidor dentro de un contenedor Docker en una instancia de la nube. Una vez hecho esto, el modelo generado puede ser guardado como un archivo y utilizado en la aplicación.

Capítulo 7 - La AI en las pruebas

Palabras clave: Automatización de pruebas

LO-7.1.1	K2	Explicar cómo la AI puede ayudar en diversas actividades de prueba: planificación y estimación de pruebas, análisis de riesgos, diseño de casos de prueba y generación de datos de prueba, asignación de defectos, análisis de impacto y análisis de cobertura.
LO-7.1.2	K2	Explicar cómo la AI puede ayudar en la elaboración de informes y cuadros de mando inteligentes.
LO-7.2.1	K2	Ilustrar el uso de varias herramientas de automatización de la ejecución de pruebas basadas en la AI disponibles en el mercado.
HO-7.2.1	H2	Utilizar las versiones demo/prueba de las herramientas de automatización de la ejecución de pruebas basadas en la AI.

La AI puede ser utilizada para mejorar los procesos existentes de pruebas en SDLC. La idea es utilizar los datos generados en los procesos de prueba para proporcionar un análisis detallado, más automatización, perspectivas y patrones más profundos y la capacidad de predecir y tomar acciones correctivas.

7.1 AI para el Ciclo de Vida de Pruebas de Software (Software Testing Life Cycle, STLC)

7.1.1 AI para métodos STLC

LO-7.1.1	K2	Explicar cómo la AI puede ayudar en diversas actividades de prueba: planificación y estimación de pruebas, análisis de riesgos, diseño de casos de prueba y generación de datos de prueba, asignación de defectos, análisis de impacto y análisis de cobertura.

En el contexto de la planificación y estimación de pruebas, el coste total del proyecto de software puede ser aproximado, teniendo en cuenta las diferentes entradas relacionadas con los proyectos de AI, incluyendo el tamaño de los

datos, el esfuerzo implicado, la elección de la plataforma, el tipo de aplicación, el tiempo de preparación de los datos, el tiempo de formación y el tiempo de prueba. Dados los datos históricos de estas entradas, se pueden preparar modelos de ML para dar estimaciones más precisas.

La AI puede ayudar a priorizar los riesgos, obtener métricas más precisas asociadas con el cumplimiento de la programación y ayudar a identificar las métricas de rendimiento de las aplicaciones con mayor precisión.

Para el diseño de pruebas, el uso de tecnologías AI, como el procesamiento de lenguaje natural (Natural language processing, NLP) y la minería de texto, puede ayudar a la generación automatizada de casos de prueba a partir de documentos de requisitos de texto. Además, la AI aplicada al análisis de código (tanto estático como dinámico), junto con el análisis de los datos recopilados de las pruebas, puede señalar problemas potenciales de rendimiento y otros requisitos no funcionales. Por otra parte, el uso de ML en datos pasados puede ayudar a identificar patrones de datos de prueba y ayuda a generar datos de prueba automatizados tanto para pruebas de componentes como para pruebas de sistemas.

En particular, para los datos de imagen y los elementos GUI, la AI puede ayudar a identificar automáticamente los elementos renderizados incorrectamente. Además, mediante un análisis centralizado de datos basado en ML de los diferentes flujos posibles, se pueden automatizar los flujos de datos correctos.

Para la predicción automatizada de defectos, los modelos que utilizan ML pueden predecir defectos basándose en métricas de calidad de código.

El análisis de impacto mediante el uso de ML en el código puede ayudar a automatizar la identificación de los módulos y archivos afectados en función del cambio.

En términos de análisis de cobertura, el uso de la AI puede ayudar a lograr una cobertura completa de pruebas y códigos mediante el análisis de los flujos de datos capturados.

7.1.2 AI para informes y cuadros de mando inteligentes

LO-7.1.2	K2	Explicar cómo la AI puede ayudar en la elaboración de informes y cuadros de mando inteligentes.

En el contexto de la presentación de informes y cuadros de mando, el uso de ML ayuda a generar perspectivas centradas y datos resumidos para su presentación en cuadros de mando inteligentes, en lugar de pura analítica.

7.2 Herramientas de automatización basadas en la AI

7.2.1 Herramientas

LO-7.2.1	K2	Ilustrar el uso de varias herramientas de automatización de la ejecución de pruebas basadas en la AI disponibles en el mercado.
HO-7.2.1	H2	Hacer uso de las versiones demo/prueba de las herramientas de automatización de la ejecución de pruebas basadas en la AI.

Existe un gran número de herramientas de automatización disponibles en el mercado. Algunas de estas herramientas utilizan la AI para hacer que la automatización sea más fácil o más mantenible

Las herramientas de AI podrían utilizar arañas GUI que atraviesan la GUI completa y registran la aplicación. A través de iteraciones, son capaces de aprender, comparar e identificar errores.

Algunas herramientas combinan elementos de pruebas visuales GUI y usan ML para averiguar los cambios y la posible correlación entre localizadores y elementos y si un cambio es un error o un cambio esperado.

Algunas herramientas utilizan la NLP, otras trabajan a nivel de DOM, otras en la interfaz de usuario, otras van al registro (log), otras combinan pruebas funcionales y de rendimiento y otras combinan algunos de los enfoques mencionados anteriormente.

Las herramientas basadas en imágenes pueden utilizar herramientas de

clasificación de imágenes basadas en AI para señalar defectos de interfaz de usuario. Estos pueden funcionar como un plug-in de navegador web con grabación y reproducción. La comparación de imágenes basada en la AI puede ser superior a la comparación de imágenes simple.

Referencias

[AG1] Neural Networks and Deep Learning - By Aurélien Géron (O'Reilly)

[BT1] A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives - By L. Anderson, P. W. Airasian, and D. R. Krathwohl (Allyn & Bacon 2001)

[BT2]
https://www.apu.edu/live_data/files/333/blooms_taxonomy_action_verbs.pdf

[DA1] 4 types of data analytics to improve decision-making
https://www.scnsoft.com/blog/4-types-of-data-analytics

[DI1] Introduction to k-Nearest Neighbors
https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/

[DSC1] https://www.datasciencecentral.com/profiles/blogs/the-data-science-project-lifecycle

[DX1] DeepXplore: Automated Whitebox Testing of Deep Learning Systems, SOSP '17 Proceedings of the 26th Symposium on Operating Systems Principles (Pages 1-18) http://www.cs.columbia.edu/~junfeng/papers/deepxplore-sosp17.pdf

[EA1] Interpretable Machine Learning: A Guide for Making Black Box Models Explainable (Section 2.1 Importance of Interpretability) - By Christoph Molnar
https://christophm.github.io/interpretable-ml-book/agnostic.html

[EA2] "Why Should I Trust You?" Explaining the Predictions of Any Classifier
https://arxiv.org/pdf/1602.04938v1.pdf

[EA3] Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization
https://arxiv.org/pdf/1610.02391.pdf

[ISTQB-FL 2018] ISTQB Foundation Level Syllabus version 2018. Available at
https://www.istqb.org/downloads/category/51-ctfl2018.html

[JB1] Machine Learning: Hands-On for Developers and Technical Professionals - By Jason Bell (WILEY 2014)

[JB2] A Gentle Introduction to Concept Drift in Machine Learning
https://machinelearningmastery.com/gentle-introduction-concept-drift-machine-learning

[KUR] The Age of Intelligent Machines - By Ray Kurzweil (MIT Press 1990)

[LA1] Visual Modeling for Test Idea Generation - Vipul Kocher
https://www.testnet.org/testnet/download/preview-voorjaar-2015/visual-modeling-for-test-idea-generation-v2.pdf

[LA2] Vibhakti - By Ujjwol Lamichhane
https://www.scribd.com/doc/30377592/Vibhakti-%E0%A4%B5%E0%A4%BF%E0%A4%AD%E0%A4%95-%E0%A4%A4%E0%A4%BF

[SMU]The CRISP-DM User Guide
https://s2.smu.edu/~mhd/8331f03/crisp.pdf

[TDA] Testing in the digital age: AI makes the difference - By Tom van de Ven, Rik Marselis and Humayun Shaukat (Sogetibooks 2018)

[UDI] Image Preprocessing
https://towardsdatascience.com/image-pre-processing-c1aec0be3edf

[UDT] Text Preprocessing in Python: Steps, Tools, and Examples
https://medium.com/@datamonsters/text-preprocessing-in-python-steps-tools-and-examples-bf025f872908

[ULA] Unsupervised Learning: Association Rules - By Krzysztof J. Cios, Roman W. Swiniarski, Witold Pedrycz, Lukasz A. Kurgan
[Wiki1] https://en.wikipedia.org/wiki/Training,_validation,_and_test_sets

[Wiki2] Random forest
https://en.wikipedia.org/wiki/Random_forest

[Wiki3] Decision tree learning
https://en.wikipedia.org/wiki/Decision_tree_learning

[Wiki4] Linear regression
https://en.wikipedia.org/wiki/Linear_regression

[Wiki5] Metamorphic testing
https://en.wikipedia.org/wiki/Metamorphic_testing